**ГБПОУ КМБ №4**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ ГОРОДА МОСКВЫ**

**«КОЛЛЕДЖ МАЛОГО БИЗНЕСА №4»**

**РАБОТУ ВЫПОЛНИЛ ОБУЧАЮЩИЙСЯ ГРУППЫ ИПО 21.23 ГОЛИШЕВСКИЙ ГЕОРГИЙ**

07.11.2024г

**«ОСНОВНЫЕ ИНСТРУМЕНТЫ WEB РАЗРАБОТКИ ПРОГРАМНОГО ОБЕСПЕЧЕНИЯ»**

1. Языки программирования  
-**HTML (Hypertext Markup Language)**    
HTML — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере. Веб-браузеры получают HTML документ от сервера по протоколам HTTP/HTTPS или открывают с локального диска, далее интерпретируют код в интерфейс, который будет отображаться на экране монитора.  
  
-**CSS (Cascading Style Sheets)**    
Таблица веб-стилей — это форма разделения содержимого и представления для веб-дизайна, в которой разметка веб-страницы содержит семантическое содержимое и структуру страницы, но не определяет ее визуальный макет.  
  
-**JavaScript**

JavaScript — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript. JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений.  
  
  
-**PHP**    
PHP — C-подобный скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.  
  
-**Python**    
Python — мультипарадигмальный высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ  
  
-**Ruby**    
Ruby — динамический, рефлективный, интерпретируемый высокоуровневый язык программирования. Язык обладает независимой от операционной системы реализацией многопоточности, сильной динамической типизацией, сборщиком мусора и многими другими возможностями.  
-**TypeScript**    
TypeScript — язык программирования, представленный Microsoft в 2012 году и позиционируемый как средство разработки веб-приложений, расширяющее возможности JavaScript. Разработчиком языка TypeScript является Андерс Хейлсберг, создавший ранее Turbo Pascal, Delphi и C#.  
  
 2. Фреймворки и библиотеки  
-**React**    
React — JavaScript-библиотека с открытым исходным кодом для разработки пользовательских интерфейсов. React разрабатывается и поддерживается Facebook, Instagram и сообществом отдельных разработчиков и корпораций. React может использоваться для разработки одностраничных и мобильных приложений.  
  
-**Vue.js**    
Vue.js — JavaScript-фреймворк с открытым исходным кодом для создания пользовательских интерфейсов. Легко интегрируется в проекты с использованием других JavaScript-библиотек. Может функционировать как веб-фреймворк для разработки одностраничных приложений в реактивном стиле.  
  
-**Angular**    
Angular — открытая и свободная платформа для разработки веб-приложений, написанная на языке TypeScript, разрабатываемая командой из компании Google, а также сообществом разработчиков из различных компаний. Angular — полностью переписанный фреймворк от той же команды, которая написала AngularJS.  
  
-**Django**    
Django — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. Проект поддерживается организацией Django Software Foundation. Сайт на Django строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми.  
  
-**Flask**    
Flask — фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2. Относится к категории так называемых микрофреймворков — минималистичных каркасов веб-приложений, сознательно предоставляющих лишь самые базовые возможности.  
  
-**Ruby on Rails**    
Ruby on Rails — фреймворк, написанный на языке программирования Ruby, реализует архитектурный шаблон Model-View-Controller для веб-приложений, а также обеспечивает их интеграцию с веб-сервером и сервером баз данных. Является открытым программным обеспечением и распространяется под лицензией MIT.  
-**Express.js**    
Express.js, или просто Express — фреймворк web-приложений для Node.js, реализованный как свободное и открытое программное обеспечение под лицензией MIT. Он спроектирован для создания веб-приложений и API. Де-факто является стандартным каркасом для Node.js.

3. Системы управления базами данных (СУБД)  
-**MySQL**    
MySQL — свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB.  
  
-**PostgreSQL**    
PostgreSQL — свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows.  
  
-**MongoDB**    
MongoDB — документоориентированная система управления базами данных, не требующая описания схемы таблиц. Считается одним из классических примеров NoSQL-систем, использует JSON-подобные документы и схему базы данных. Написана на языке C++.  
  
-**SQLite**    
SQLite — компактная встраиваемая СУБД. Исходный код библиотеки передан в общественное достояние. В 2005 году проект получил награду Google-O’Reilly Open Source Awards.

4. Инструменты разработки и IDE  
-**Visual Studio Code (VS Code)**    
Visual Studio Code — редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений.  
-**JetBrains WebStorm**    
JetBrains WebStorm — интегрированная среда разработки на JavaScript, CSS & HTML от компании JetBrains, разработанная на основе платформы IntelliJ IDEA. WebStorm обеспечивает автодополнение, анализ кода на лету, навигацию по коду, рефакторинг, отладку, и интеграцию с системами управления версиями.  
-**Sublime Text**    
Sublime Text — проприетарный текстовый редактор. Поддерживает плагины на языке программирования Python. Разработчик позволяет бесплатно и без ограничений ознакомиться с продуктом, однако программа уведомляет о необходимости приобретения лицензии.  
  
-**Atom**    
Atom — бесплатный текстовый редактор с открытым исходным кодом для Linux, macOS, Windows с поддержкой плагинов, написанных на JavaScript, и встраиваемых под управлением Git. Большинство плагинов имеют статус свободного программного обеспечения, разрабатываются и поддерживаются сообществом.

5. Системы контроля версий  
-**Git**    
Git — распределённая система управления версиями. Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux, первая версия выпущена 7 апреля 2005 года; координатор — Дзюн Хамано.

Git — это система контроля версий, предназначенная для отслеживания изменений в коде и управления версиями программного обеспечения. Git позволяет многим разработчикам работать одновременно над одним проектом, не мешая друг другу, и эффективно управлять изменениями в коде. Вот подробнее о его ключевых особенностях и принципах работы:

1. Основные понятия Git - Репозиторий (repository): Это база данных, в которой хранятся все изменения в проекте, а также информация о том, кто и когда их вносил. - Коммиты (commits): Это снимки состояния вашего проекта в определённый момент времени. Каждый коммит содержит информацию о том, какие изменения были внесены, а также метаданные о пользователе и времени изменений. - Ветки (branches): Ветки позволяют разработчикам работать над разными частями проекта параллельно. Основная ветка обычно называется main или master, но вы можете создавать новые ветки для разработки новых функций или исправления багов. - Слияния (merge): Когда работа в разных ветках завершена, их изменения можно объединить в одну ветку с помощью слияния. - Конфликты слияния (merge conflicts): Иногда два разработчика могут изменить одни и те же строки кода, что приводит к конфликтам при слиянии веток. Git не может автоматически разрешить такие конфликты, и разработчик должен вручную выбрать, какие изменения оставить.

2. Как работает Git - Локальные и удалённые репозитории: Git позволяет работать с репозиториями как локально (на вашем компьютере), так и удалённо (например, на GitHub, GitLab или Bitbucket). Вы можете изменять код локально, а затем синхронизировать изменения с удалённым репозиторием с помощью команд git push и git pull. - Цикл работы с Git:

1. Создание коммита: Вы изменяете код, добавляете файлы в индекс с помощью команды git add и затем сохраняете изменения в репозитории с помощью git commit.

2. Обновление удалённого репозитория: Чтобы поделиться своими изменениями с другими разработчиками, вы используете команду git push.

3. Получение изменений: Чтобы забрать изменения, сделанные другими участниками проекта, используйте команду git pull.

3. Команды Git - `git init`: Инициализация нового репозитория. - `git clone <url>`: Клонирование удалённого репозитория на ваш локальный компьютер. - `git status`: Просмотр текущего состояния репозитория (например, какие файлы были изменены или добавлены). - `git add <file>`: Добавление файлов в индекс (готовность к коммиту). - `git commit -m "message"`: Сохранение изменений с сообщением. - `git push`: Отправка изменений в удалённый репозиторий. - `git pull`: Забор изменений из удалённого репозитория. - `git branch`: Просмотр веток репозитория или создание новой ветки. - `git merge <branch>`: Слияние одной ветки с другой. - `git log`: Просмотр истории коммитов.

4. Преимущества Git - Распределённая архитектура: В отличие от централизованных систем контроля версий, Git является распределённой. Это означает, что каждый разработчик работает с полным копией репозитория, и не требуется постоянное подключение к серверу для работы. - Быстродействие: Git очень быстрый, так как многие операции выполняются локально, например, просмотр истории, создание веток, слияние и другие. - Гибкость и мощь: Git предоставляет широкий набор возможностей для управления ветками, разрешения конфликтов, работы с подмодулями и другими задачами. - Поддержка командной работы: Git делает сотрудничество удобным. Возможности для слияния веток и истории изменений позволяют эффективно отслеживать работу разных людей и минимизировать ошибки.

5. Потоки работы (workflow) Существует несколько подходов к организации работы в Git. Вот самые популярные: - Git flow: Это методология, при которой разработка ведётся в нескольких ветках: feature (новые фичи), develop (разработка), release (подготовка к релизу) и master (стабильные версии). - GitHub flow: Это более упрощённый подход, используемый, например, на GitHub. Работает с одной основной веткой (`main`), а все изменения выполняются через ветки с Pull Requests (PR).

-**GitHub**

GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. Веб-сервис основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc.

GitHub — это веб-сервис для хостинга и управления проектами, использующий систему контроля версий Git. Он позволяет разработчикам и командам совместно работать над кодом, отслеживать изменения и автоматизировать процессы разработки. Вот более подробное описание его ключевых функций:

1.**Система контроля версий Git**GitHub основан на Git, распределённой системе контроля версий, которая позволяет разработчикам отслеживать изменения в коде, управлять различными версиями и легко объединять изменения, сделанные разными людьми. GitHub упрощает работу с Git, предоставляя удобный интерфейс для выполнения операций, таких как коммиты, ветвление, слияние и другие.  
  
 2.**Репозитории (Repositories)**Каждый проект на GitHub хранится в репозитории. Репозиторий — это структура, в которой содержится весь код, файлы и история изменений. Репозитории могут быть публичными (доступными всем) или приватными (доступными только выбранным пользователям). Разработчики могут легко клонировать репозитории, вносить изменения и отправлять их обратно с помощью pull-запросов.  
  
 3.**Pull-запросы (Pull Requests)**Pull-запрос — это запрос на вливание изменений из одной ветки (или форка) в основную ветку (обычно main или `master`). Это важный инструмент для совместной работы, поскольку он позволяет другим участникам проекта проверить предложенные изменения перед их объединением. В pull-запросах можно обсуждать код, оставлять комментарии и вносить корректировки.  
  
 4.**Issues и Project Management**GitHub предоставляет инструменты для отслеживания задач (issues), которые могут быть использованы для поиска и устранения ошибок, добавления новых функций и планирования работы над проектом. Также есть возможность организовать задачи в виде канбан-досок через GitHub Projects.  
  
 5.**GitHub Actions**GitHub Actions — это инструмент для автоматизации рабочего процесса (CI/CD), который позволяет запускать скрипты и тесты при определённых условиях. Например, можно настроить автоматический запуск тестов при каждом коммите или сборку приложения при слиянии в основную ветку.  
  
 6.**Fork и Collaboration**Forking (форк) — это процесс создания копии репозитория. Это особенно полезно для работы с проектами с открытым исходным кодом. Пользователи могут форкать репозиторий, вносить изменения в свою копию и предлагать их оригинальному проекту через pull-запросы. Этот процесс делает совместную разработку более гибкой и контролируемой.  
  
 7.**GitHub Pages**GitHub Pages позволяет бесплатно хостить веб-сайты прямо из репозитория. Это может быть полезно для хостинга личных сайтов, блогов или документации для проектов. Поддерживает статические страницы, написанные с использованием HTML, CSS и JavaScript.  
  
 8.**Сообщество и Open Source**GitHub также является важной платформой для open-source (открытого исходного кода). Множество известных проектов, таких как Linux, React, TensorFlow и другие, размещены на GitHub. Пользователи могут вносить свои изменения в проекты, создавать и обсуждать баг-репорты, делиться идеями.  
  
 9.**Интеграции и Облачные сервисы**GitHub поддерживает множество интеграций с различными инструментами, такими как Slack, Trello, Jenkins, Docker, и другими сервисами для разработки. Это позволяет создавать полноценные решения для автоматизации разработки, тестирования и деплоя.  
  
 10.**GitHub Copilot**GitHub Copilot — это интеллектуальный помощник для разработчиков, использующий технологии искусственного интеллекта. Он предлагает автодополнения и подсказки для кода, что ускоряет процесс разработки.  
  
GitHub имеет как бесплатные, так и платные планы. Бесплатная версия предоставляет доступ ко всем основным функциям, включая публичные репозитории и некоторые возможности для приватных репозиториев.  
  
В целом, GitHub — это мощная и универсальная платформа для разработчиков, которая позволяет не только управлять кодом, но и поддерживать активное сотрудничество, автоматизировать процессы и работать с открытым исходным кодом.

**- GitLab**

GitLab — платформа для совместной разработки. Она позволяет командам организовывать свои процессы от планирования до мониторинга и может работать в облаке.

GitLab — это платформа для DevOps, которая предлагает инструменты для управления проектами, контроля версий, CI/CD (непрерывной интеграции и непрерывного деплоя), а также для тестирования и мониторинга. GitLab предоставляет полный цикл разработки программного обеспечения, объединяя управление репозиториями Git с инструментами для автоматизации, планирования и мониторинга.  
  
Вот более подробное описание ключевых возможностей и особенностей GitLab:  
  
 1.**Git-репозитории**Как и GitHub, GitLab использует систему контроля версий Git для управления кодом. Репозитории могут быть публичными или приватными, и позволяют пользователям следить за изменениями в коде, создавать ветки, работать с pull-запросами (или Merge Request в GitLab) и сливать их в основную ветку.  
  
 2.**Merge Requests (MR)**Merge Requests (или pull-запросы в других системах) — это механизмы, с помощью которых разработчики могут предложить изменения в коде, чтобы они были рассмотрены и интегрированы в основной репозиторий. В GitLab можно не только обсуждать и проверять изменения, но и запускать автоматические тесты или проверки качества кода перед слиянием.  
  
 3.**CI/CD (непрерывная интеграция и деплой)**GitLab предоставляет встроенную систему CI/CD для автоматизации сборки, тестирования и деплоя приложений. Это одна из самых сильных сторон платформы. GitLab CI/CD позволяет настроить автоматический процесс, который запускается на основе событий (например, после коммита или при создании pull-запроса). Конфигурация CI/CD в GitLab производится через файл .gitlab-ci.yml, в котором описывается, какие этапы должны выполняться, какие тесты — запускаться, а какие действия — выполняться при деплое.  
  
 4.**GitLab Runner**GitLab Runner — это инструмент, который выполняет задачи CI/CD. Он может быть настроен для работы на различных средах (например, Docker, виртуальные машины, физические серверы и т.д.), обеспечивая гибкость и масштабируемость. GitLab Runner запускает задания, такие как сборка, тестирование или деплой, согласно настройкам в .gitlab-ci.yml.  
  
 5.**Автоматизация и Обсуждения**GitLab также предлагает функции для управления задачами и отслеживания багов через систему**Issues.** Вопросы могут быть связаны с коммитами, merge request'ами и даже с другими проблемами, создавая прозрачный и удобный процесс разработки. Возможности для комментирования и обсуждения изменений в реальном времени делают GitLab отличным инструментом для командной работы.  
  
 6.**GitLab Auto DevOps**Auto DevOps — это набор стандартных настроек CI/CD для быстрой автоматизации процессов разработки, тестирования и деплоя. Этот инструмент облегчает настройку CI/CD, предоставляя дефолтные конфигурации для самых популярных технологий и языков. Это позволяет новичкам начать работу с DevOps практиками без необходимости ручной настройки каждого этапа.  
  
### 7.**Безопасность и качественный код**GitLab предлагает инструменты для обеспечения безопасности и качества кода:  
-**Static Application Security Testing (SAST):** анализирует код на предмет уязвимостей безопасности.  
-**Dynamic Application Security Testing (DAST):** анализирует приложение в работающем состоянии для поиска уязвимостей.  
-**Dependency Scanning:** проверяет зависимости на наличие уязвимостей.  
-**Container Scanning:** анализирует контейнеры Docker на безопасность.  
  
GitLab также интегрируется с различными инструментами для обеспечения качества кода, такими как линтеры, тесты покрытия и другие.  
  
 8.**GitLab Pages**GitLab Pages позволяет пользователям хостить статические сайты, как и GitHub Pages, но с дополнительными возможностями интеграции и автоматизации. GitLab Pages поддерживает различные типы статических сайтов (например, на Jekyll, Hugo или других генераторах), а также предоставляет возможность для деплоя через CI/CD пайплайны.  
  
 9.**Интерфейс и интеграции**

GitLab предлагает богатый пользовательский интерфейс для работы с репозиториями, задачами, пайплайнами и отчетами. Он также поддерживает интеграции с другими популярными сервисами, такими как Jira, Slack, Kubernetes и другими.

**10. Управление проектами**

GitLab предоставляет удобные инструменты для управления проектами. Например, GitLab Boards позволяет создавать и управлять задачами через интерфейс Kanban, что делает планирование и распределение работы удобным и эффективным.

**11. Самостоятельный хостинг (Self-hosted)**

Одной из уникальных особенностей GitLab является возможность самостоятельного хостинга. Вы можете установить GitLab на собственные серверы, чтобы иметь полный контроль над всеми данными, настройками безопасности и доступом. Это особенно важно для крупных организаций, которые не хотят размещать свой код на сторонних сервисах.

**12. Гибкость тарифных планов**

GitLab предлагает как облачные решения (SaaS), так и возможность развертывания на собственных серверах (Self-Managed). Он имеет несколько тарифных планов, включая бесплатный (для небольших команд и проектов), а также платные версии с расширенными возможностями для крупных организаций, таких как дополнительные функции безопасности, масштабируемость и поддержку.

**13. Open Source и сообщества**

GitLab имеет открытый исходный код (за исключением некоторых корпоративных функций), что позволяет сообществу вносить вклад в улучшение платформы. С момента своего создания GitLab активно развивался благодаря открытым разработчикам, и это стало основой для его успеха в мире DevOps.

**- Bitbucket**    
Это инструмент для хостинга кода и совместной работы на основе Git, предназначенный для команд. Лучшие в своем классе интеграции Jira и Trello для Bitbucket создают для всей команды разработчиков единое пространство, в котором ее участники вместе работают над проектом.

6. Инструменты для сборки и автоматизации  
-**Webpack**    
  
  
-**Gulp**    
Gulp — таск-менеджер для автоматического выполнения часто используемых задач, написанный на языке программирования JavaScript. Программное обеспечение использует командную строку для запуска задач, определённых в файле Gulpfile. Создан как ответвление от проекта Grunt, чтоб взять из него лучшие практики.  
  
-**Grunt**    
Grunt — менеджер задач для автоматического выполнения рутинных операций, написанный на языке программирования JavaScript. Программное обеспечение использует командную строку для запуска задач, определённых в файле Gruntfile. Распространяется через менеджер пакетов NPM под лицензией MIT.

7. Протоколы и технологии  
-**HTTP/HTTPS**    
HTTPS — расширение протокола HTTP для поддержки шифрования в целях повышения безопасности. Данные в протоколе HTTPS передаются поверх криптографических протоколов TLS или устаревшего в 2015 году SSL. В отличие от HTTP с TCP-портом 80, для HTTPS по умолчанию используется TCP-порт 443.

HTTPS означает «Безопасный протокол передачи гипертекста». Это расширение HTTP (протокола передачи гипертекста) и используется для безопасной связи через компьютерную сеть, в первую очередь через Интернет. HTTPS гарантирует, что данные, которыми обмениваются браузер пользователя и веб-сайт, зашифрованы, что означает, что третьи лица не смогут легко их перехватить. Ключевые особенности HTTPS включают в себя:

1. Шифрование.

Это гарантирует, что данные, передаваемые между вашим браузером и веб-сайтом, будут нечитаемы для тех, кто может попытаться их перехватить. Обычно это достигается с помощью таких протоколов, как SSL (Secure Sockets Layer) или TLS (Transport Layer Security).

1. Аутентификация. HTTPS проверяет, что веб-сайт, с которым вы общаетесь, является законным сайтом, а не мошенником. Это помогает защититься от атак «человек посередине».
2. Целостность данных. HTTPS гарантирует, что данные, передаваемые между вашим браузером и веб-сайтом, не были изменены при передаче. Чтобы идентифицировать веб-сайт с поддержкой HTTPS, вы увидите «https://» в URL-адресе, а в большинстве современных браузеров рядом с адресной строкой отображается значок замка, указывающий на безопасное соединение.  
   -**REST API**    
   REST — архитектурный стиль взаимодействия компонентов распределённого приложения в сети. Другими словами, REST — это набор правил того, как программисту организовать написание кода серверного приложения, чтобы все системы легко обменивались данными и приложение можно было масштабировать.  
     
   -**GraphQL**    
   GraphQL — язык запросов данных и язык манипулирования данными с открытым исходным кодом для построения веб ориентированных программных интерфейсов. GraphQL был разработан как внутренний проект компании Facebook в 2012 году, а позднее в 2015 году был выпущен публично.  
     
   -**WebSockets**    
   WebSocket — протокол связи поверх TCP-соединения, предназначенный для обмена сообщениями между браузером и веб-сервером, используя постоянное соединение. В настоящее время в W3C осуществляется стандартизация API Web Sockets. Черновой вариант стандарта этого протокола утверждён IETF.

8. Инструменты для тестирования  
-**Jest**    
Обеспечивая вашим тестам уникальное глобальное состояние, Jest добивается безошибочного запуска тестов в параллельных потоках. Чтобы делать это быстро, Jest в первую очередь запускает проваленные ранее тесты, и меняет очередность их запуска, отталкиваясь от того, как долго выполняется каждый тест.  
  
-**Mocha**    
Mocha — фреймворк для тестирования на языке JavaScript. Функционирует на основе Node.js, поддерживает работу с браузерами, асинхронное тестирование, отчеты о покрытия продукта тестами, а также работу с любой «ассертной» библиотекой.  
  
-**Cypress**

Cypress — это инструмент автоматизации внешнего тестирования для регрессионного тестирования веб-приложений. Cypress работает на Windows, Linux и macOS. Приложение Cypress — это программное обеспечение с открытым исходным кодом, выпущенное по лицензии MIT, а Cypress Cloud — это веб-приложение.  
  
-**Selenium**    
Selenium WebDriver — это инструмент для автоматизации действий веб-браузера. В большинстве случаев используется для тестирования Web-приложений, но этим не ограничивается. В частности, он может быть использован для решения рутинных задач администрирования сайта или регулярного получения данных из различных источников.

9. Облачные сервисы и хостинг  
-**AWS (Amazon Web Services)**    
Amazon Web Services — коммерческое публичное облако, поддерживаемое и развиваемое компанией Amazon с 2006 года. Предоставляет подписчикам услуги как по инфраструктурной модели, так и платформенного уровня.  
  
-**Google Cloud Platform (GCP)**    
Google Cloud Platform — предоставляемый компанией Google набор облачных служб, которые выполняются на той же самой инфраструктуре, которую Google использует для своих продуктов, предназначенных для конечных потребителей, таких как Google Search и YouTube.  
  
-**Netlify**   
Netlify — это компания, специализирующаяся на удаленных облачных вычислениях, которая предлагает платформу разработки, включающую в себя создание, развертывание и бессерверные серверные службы для веб-приложений и динамических веб-сайтов.  
  
-**Heroku**    
Heroku — облачная PaaS-платформа, поддерживающая ряд языков программирования. С 2010 года является дочерней компанией Salesforce.com.

10. Управление проектами  
-**Trello**

Trello — облачная программа для управления проектами небольших групп, разработанная Fog Creek Software. Trello использует парадигму для управления проектами, известную как канбан, метод, который первоначально был популяризирован Toyota в 1980-х для управления цепочками поставок.  
  
-**Asana**

Asana, Inc. — американская компания-разработчик программного обеспечения, базирующаяся в Сан-Франциско, чей флагманский сервис Asana представляет собой веб- и мобильную платформу «управления работой», предназначенную для помощи командам в организации, отслеживании и управлении своей работой.

**- ClickUp**    
ClickUp — это программное обеспечение для управления проектами с такими функциями, как задачи, виртуальные доски, электронные таблицы и совместная работа над документами на единой платформе. Ее основали Зеб Эванс и Алекс Юрковски в 2017 году.